

AMENDMENTS TO CLAIMS

1. **(Currently Amended)** A method for bootstrapping a processor from a volatile memory device connected to the processor, comprising the steps of:
bootstrapping a first processor from flash device;
asserting reset lines of the second processor;
loading boot code for the second processor from the flash device into the volatile memory device, wherein the volatile memory device is synchronous static random access memory; and
de-asserting the reset lines of the second processor,
wherein the processor performs the bootstrap procedure using the boot code stored in the volatile memory device.
2. **(Cancelled)**
3. **(Original)** The method according to claim 1, wherein a complex programmable logic device generates the reset lines of the processor.
4. **(Original)** The method according to claim 3, wherein the reset lines are controlled by the first processor and handled by the complex programmable logic device.
5. **(Cancelled)**
6. **(Original)** The method according to claim 1, wherein a plurality of processors are bootstrapped by loading the boot code for the plurality of processors into a plurality of volatile memory devices, wherein each processor is connected to a different volatile memory device.
7. **(Original)** The method according to claim 6, wherein the first processor provides identity to each of the plurality of processors by posting information through volatile memory.
8. **(Cancelled)**
9. **(Original)** The method according to claim 1, wherein the volatile memory device is dual port random access memory.
10. **(Cancelled)**
11. **(Currently Amended)** The method according to claim 8 1, wherein the volatile memory device is a synchronous dual port static random access memory.

12. (Currently Amended) The method according to claim ~~10~~ 1, wherein the synchronous static random access memory has a discontinuous clock throughout the bootstrapping procedure.
13. (Currently Amended) A system for bootstrapping a plurality of slave processors from a plurality of volatile memory devices, comprising:
 - a ~~second-master~~ processor with an associated flash device;
 - a logic device for generating reset lines of the ~~second-slave~~ processors and deasserting the reset lines based on control signals from the ~~first-master~~ processor;
 - the volatile memory devices ~~for~~ storing boot code for the ~~second~~ slave microprocessors loaded from the flash device after the ~~first-master~~ processor has been bootstrapped; ~~and~~
 - the slave processors each being connected to ~~the a separate~~ volatile memory device, wherein when the reset lines of the ~~second-slave~~ processors are deasserted, the ~~second-processor bootstraps~~ slave processors bootstrap from the boot code stored in the volatile memory devices;
 - wherein the master processor provides identity to each of the slave processors by posting information through volatile memory.
14. (Cancelled)
15. (Currently Amended) The system according to claim 13, wherein the logic ~~unit~~ device is a complex programmable logic device.
16. (Currently Amended) The system according to claim 15, wherein the logic ~~unit~~ device is a field programmable gate array.
- 17.-19. (Cancelled)
20. (Currently Amended) The system according to claim 13, wherein the volatile memory ~~device is~~ devices are static random access ~~memory~~ memories.
21. (Currently Amended) The system according to claim 13, wherein the volatile memory ~~device is~~ devices are dual port random access ~~memory~~ memories.
22. (Currently Amended) The system according to claim ~~19~~ 13, wherein the volatile memory device is a synchronous static random access memory.
23. (Currently Amended) The system according to claim ~~22~~ 13, wherein the ~~pipeline static-random-access-volatile~~ memory ~~has~~ devices have a discontinuous clock through out the bootstrapping procedure.

24. **(Currently Amended)** The system according to claim 13, further comprising:
a plurality of slave flash devices each connected to one of the second slave
processors, the slave flash devices having boot code for the slave
processors, wherein the first master processor determines whether to
bootstrap the second slave processors using the boot code in the slave
flash devices ~~connected to the second processor~~ or the boot code in the
volatile memory devices ~~based on a user selection~~.
25. **(Currently Amended)** The system according to claim 24, wherein the first master
processor sets a command register in the logic device to configure logic units for
the selected bootstrap procedure.
26. **(Currently Amended)** A method for bootstrapping a processor from either a
flash device or a volatile memory device, comprising the steps of:
bootstrapping a master processor;
determining whether a flash bootstrap procedure or a volatile memory device
bootstrap procedure has been selected for the second microprocessor;
setting a command register in a logic device to configure logic units for the
selected bootstrap procedure;
if the flash bootstrap procedure was selected:
performing bootstrap procedure from the flash device associated with the
second processor;
if the volatile memory bootstrap procedure was selected:
asserting reset lines of the processor;
loading boot code for the second processor from a flash device associated
with the first processor into the volatile memory device;
and de-asserting the reset lines of the processor, wherein the processor
bootstraps from the boot code stored in the volatile memory device
wherein the volatile memory device is a synchronous static random
access memory.
27. **(Cancelled)**
28. **(Original)** The method according to claim 26, wherein a complex programmable
logic device generates the reset lines of the second processor.
29. **(Original)** The method according to claim 28, wherein the reset lines are
controlled by the master processor.

30. (Cancelled)
31. (Original) The method according to claim 26, wherein a plurality of processors are bootstrapped by loading the boot code for the plurality of processors into a plurality of volatile memory devices, wherein each processor is connected to a different volatile memory device.
32. (Currently Amended) The method according to claim 31, wherein the master processor provides identity to each of the plurality of processors by posting information through ~~shared~~ the volatile memory devices.
33. (Cancelled)
34. (Original) The method according to claim 26, wherein the volatile memory device is dual port random access memory.
35. (Cancelled)
36. (Currently Amended) The method according to claim ~~35~~ 26, wherein the synchronous static random access memory has a discontinuous clock through out the bootstrapping procedure.
37. (Currently Amended) A mechanism for bootstrapping a processor from a volatile memory device connected to the processor, comprising the steps of:
means for bootstrapping a master processor from flash device;
means for asserting reset lines of the processor;
means for loading boot code for the processor from the flash device into the volatile memory device; and
means for de-asserting the reset lines of the processor,
wherein the processor performs the bootstrap procedure using the boot code stored in the volatile memory device and further wherein the volatile memory device is a synchronous static random access memory.
38. (Cancelled)
39. (Original) The mechanism of claim 37, wherein a complex programmable logic device generates the reset lines of the processor.
40. (Original) The mechanism of claim 37, wherein the reset lines are controlled by the master processor.
41. (Cancelled)
42. (Original) The mechanism of claim 37, wherein a plurality of processors are bootstrapped by loading the boot code for the plurality of processors into a

- plurality of volatile memory devices, wherein each processor is connected to a different volatile memory device.
43. **(Currently Amended)** The mechanism of claim 42, wherein the master processor provides identity to each of the plurality of processors by posting information through ~~shared~~ the volatile memory devices.
44. **(Cancelled)**
45. **(Original)** The mechanism of claim 37, wherein the volatile memory device is dual port random access memory.
46. **(Cancelled)**
47. **(Currently Amended)** The mechanism of claim 45, wherein the memory control signals of the dual portstatic random access memory has a discontinuous clock through out the bootstrapping procedure.
48. **(Newly Added)** A computing system comprising:
a nonvolatile memory device containing a master bootstrap process;
a synchronous static random access memory containing a slave bootstrap process;
a master processor in communication with the nonvolatile memory device, the master processor booting with the master bootstrap process, the master processor storing the slave bootstrap process on the synchronous static random access memory after the master processor has booted;
a slave processor in communication with the synchronous static random access memory, the slave processor booting with the slave bootstrap process after the slave bootstrap process has been stored on the synchronous static random access memory by the master processor.
49. **(Newly Added)** A computing system comprising:
a nonvolatile memory device containing a master bootstrap process;
a synchronous random access memory containing a slave bootstrap process;
a master processor in communication with the nonvolatile memory device, the master processor booting with the master bootstrap process;
a slave processor providing a clock signal to the synchronous random access memory;
a data bus providing data communication between the synchronous random access memory and the slave processor; and

- a pulldown resister network on the data bus.
50. **(Newly Added)** A computing system comprising:
- a first volatile memory device containing a first identity;
 - a second volatile memory device containing a second identity;
 - a master processor in communication with the nonvolatile memory devices, the master processor storing the first identity on the first volatile memory device, and the second identity on the second volatile memory device;
 - a first slave processor in communication with the first volatile memory device, the first slave processor obtaining the first identity from the first volatile memory; and
 - a second slave processor in communication with the second volatile memory device, the second slave processor obtaining the second identity from the second volatile memory.
51. **(Newly Added)** The computing system of claim 50, wherein the master processor stores on each volatile memory device a slave bootstrap process, and wherein the slave processors each boot with the slave bootstrap process received from the volatile memory devices.
52. **(Newly Added)** A method for bootstrapping a processor from a synchronous memory device connected to the processor, comprising the steps of:
- loading the synchronous memory device with a boot code;
 - resetting the processor after the boot code is loaded in the synchronous memory device;
 - submitting no-op commands to the processor until the processor provides a reliable clock signal;
 - supplying the reliable clock signal to the synchronous memory device; and
 - loading the boot code from the synchronous memory device to the processor.
53. **(Newly Added)** The method of claim 52, wherein the no-op command is created by a resister network on a data bus.
54. **(Newly Added)** The method of claim 53, wherein the resister network is a pulldown network that pulls the data bus to a zero state.
55. **(Newly Added)** The method of claim 52, wherein the synchronous memory device has a discontinuous clock during the loading of the boot code to the processor causing a period of clock instability.

56. **(Newly Added)** The method of claim 55, wherein an instruction cache of the processor is used in conjunction with the no-op commands during the period of clock instability.
57. **(Newly Added)** A computing system comprising:
- a nonvolatile memory device containing a slave bootstrap process;
 - a synchronous volatile random access memory on which the slave bootstrap process is transferred after power up;
 - a data bus;
 - a master processor providing a control signal;
 - a slave processor providing a memory control signal, receiving a reset control signal, and receiving data from the synchronous volatile random access memory over the data bus; and
 - a programmable logic device acting in response to the control signal from the master processor and the memory control signal from the slave processor, the programmable logic device providing a reset control signal to the slave processor and a memory control signal to the synchronous volatile random access memory.
58. **(Newly Added)** The system of claim 57, further comprising a clock signal from the slave processor being provided to the synchronous volatile random access memory, and a pulldown resistor network on the data bus ensuring that the slave processor receives no-op instructions over the data bus during periods of clock instability.